ITER Fast Plant System Controller Prototype Based on PXIe Platform

M. Ruiz, J. Vega, R. Castro, J.M. López, E. Barrera, G. Arcas, D. Sanz, J. Nieto, B.Gonçalves, J. Sousa, B. Carvalho, N. Utzel, P. Makijarvi

on behalf of CIEMAT/UPM/IST/ITER team

Technical University of Madrid, Spain

Asociación Euratom/CIEMAT

IPFN, Instituto Superior Técnico

ITER Organization

Outline

- Project scope and requirements.
- Alpha version
  - FPSC HW elements.
  - FPSC SW elements.
  - Results & conclusions
- Beta version
  - FPSC HW elements.
  - FPSC SW elements.
  - Results & conclusions

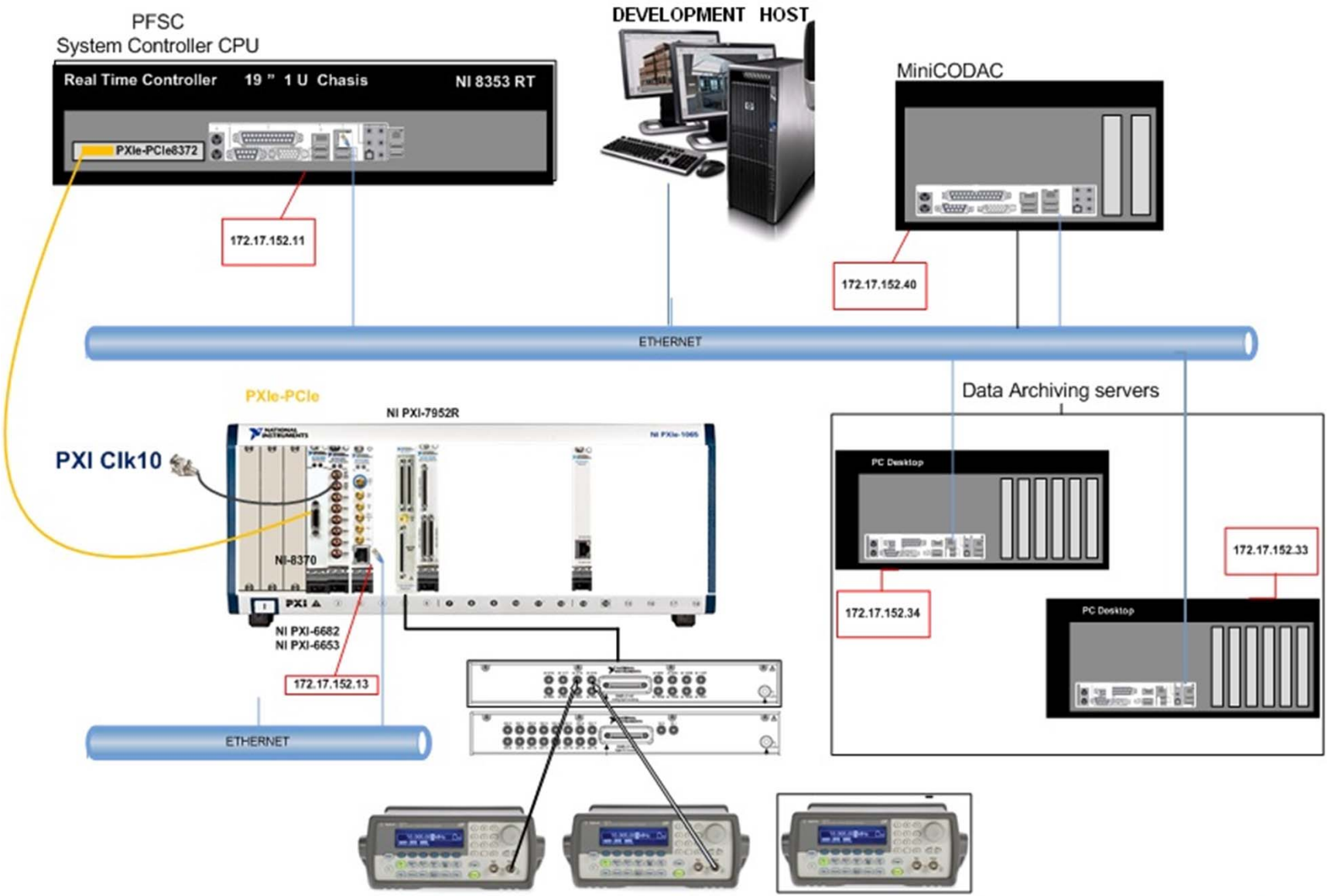
Project Scope and functional requirements

- Developing a prototype FPSC targeting Data Acquisition for ITER IO
  - Two different form factors for the implementation:
    - ATCA based solution (IST) presented by B. Goncalves (02-4)
    - PCIe based solution (CIEMAT/UPM)
- The essential "functional requirements" of FPSC to be solved in this project are:
  - To provide high rate data acquisition, preprocessing, archiving and efficient data distribution among the different FPSC software modules (functional elements).
  - To interface with the networks (PON, TCN, SDN, streaming/archiving)
  - To implement the FPSC software using RHEL and EPICS. The system setup and operation must be done using EPICS process variables and the functional elements must be interfaced using EPICS asyn layer.
  - To use COTS solutions.
- Restrictions at the project beginning
  - The device drivers, and EPICS device support (asyn interface) not available (or in development process).
  - Development to be finished in a short limited time in order to obtain figures of performance
- **Decision: a two steps approach, Alpha and Beta versions**

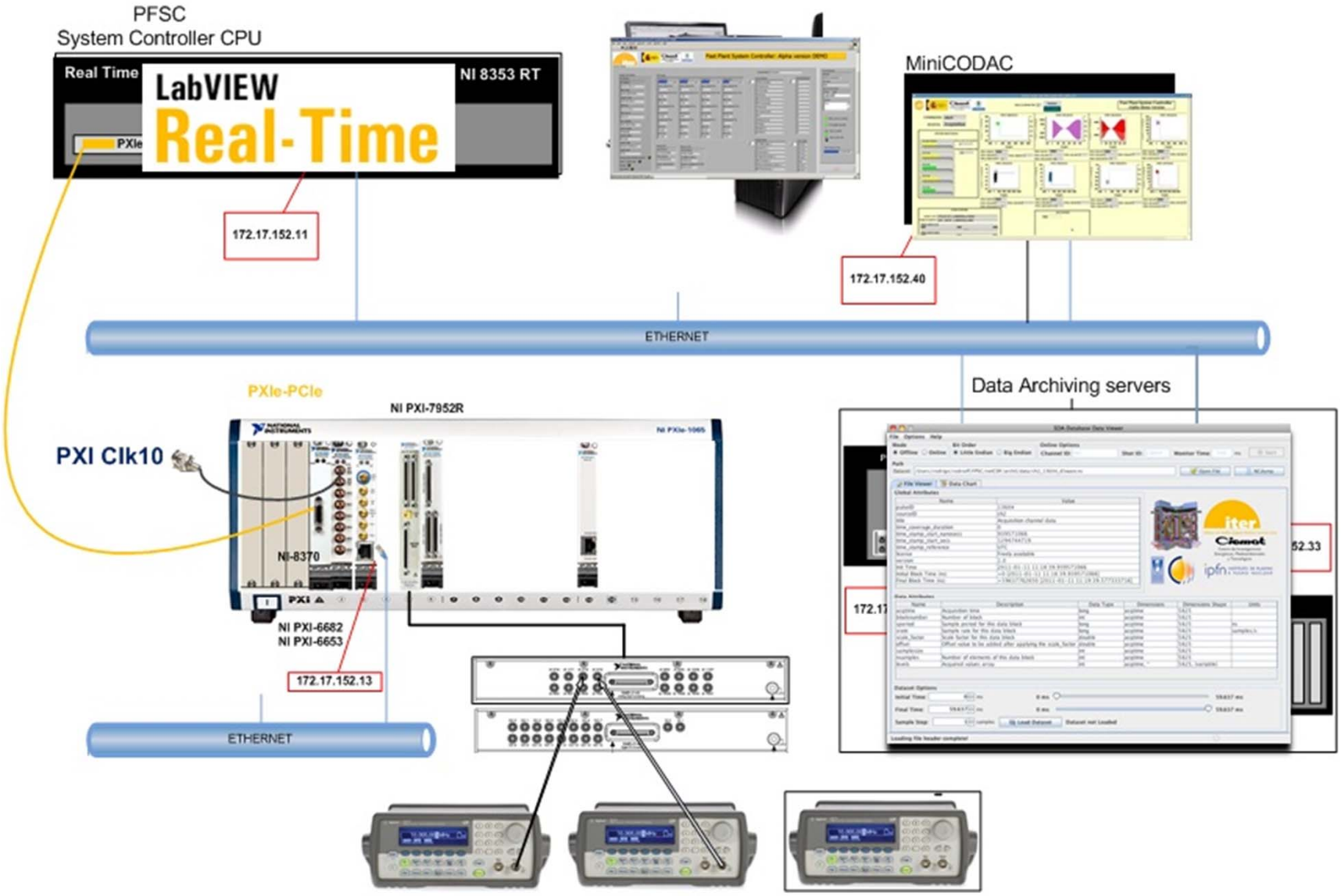
Project development: alpha version

- Implemented using:
  - Labview Real Time (to avoid third parties dependences, to test system capabilities and to learn about problems and gain experience for the beta version)
  - PXIe solution with:
    - National Instruments hardware (PXIe chassis, timing modules, DAQ using FlexRIO and an external controller)
    - LabVIEW RT Module applications running in the controller
    - LabVIEW FPGA for FlexRIO
    - LabVIEW EPICS IOC for real time target for supporting channel access.
  - Specific applications developed for running in external computers for streaming/archiving, data processing with GPUs, and monitoring using ITER CODAC Core System.

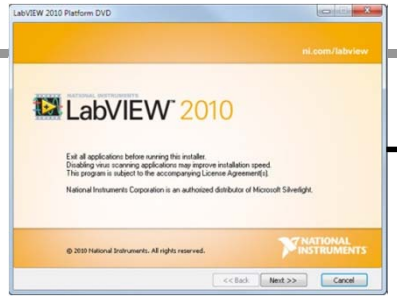
FPSC Block Diagram



FPSC Block Diagram



FPSC software elements



Development System

Signal Generator

EPICS IOC

EPICS IOC

EPICS IOC

EPICS IOC

FPSC Labview RT

FPSC GPU System

Archiver Service

Archiver Service

d1wave and event files

NFS

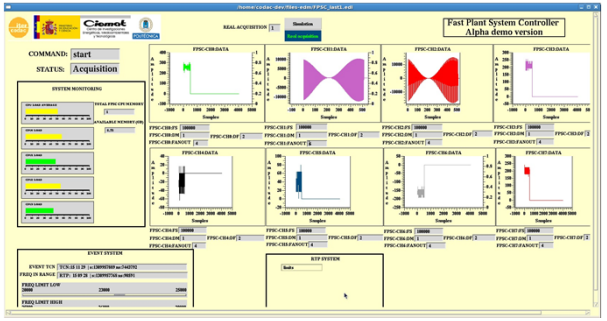
CODAC State Machine

CODAC IOC

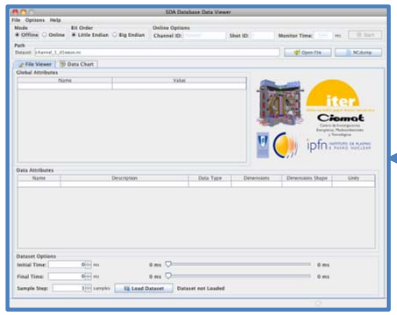
iocLog Service

Configuration Management

Pulse Control



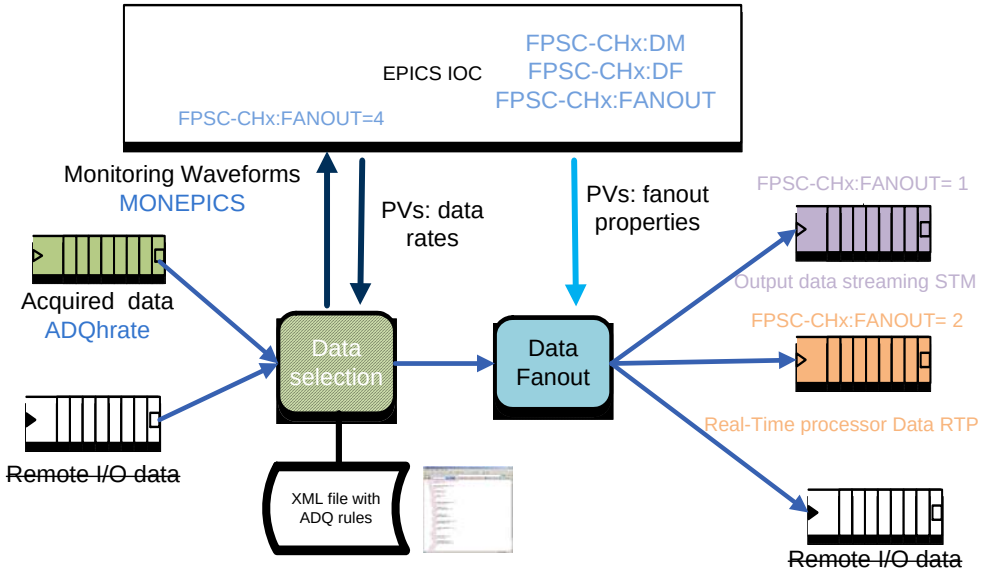
Archiving monitoring



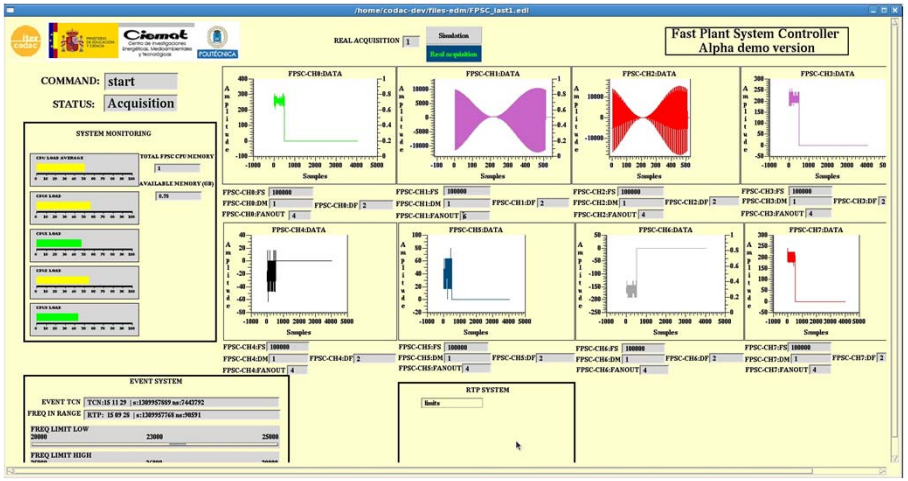
Archive Viewer

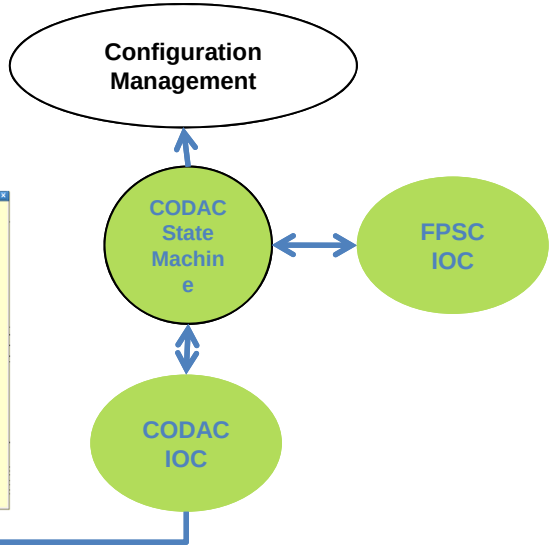
Main features of FPSC software-Alpha Version

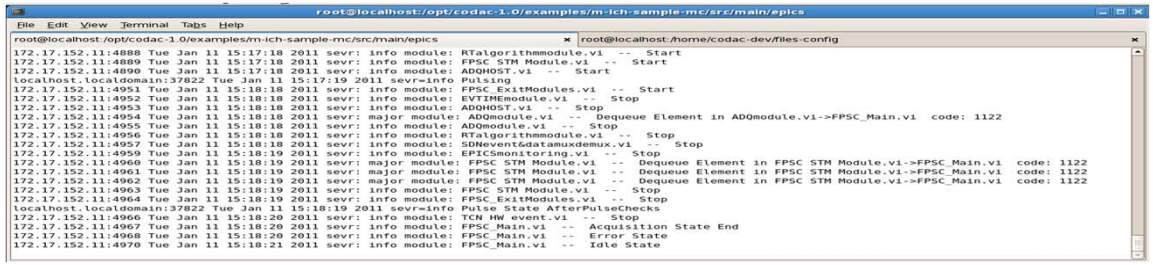
- ADQ parameters are controlled-changed using PVs (also during the pulse):
  - Sampling rate and block size for FlexRIO device.
  - Decimation factor and modes for EPICS monitoring
- FPSC State machine control and status using PVs: start/stop, memory used, CPU load, etc.
- Acquired data can be sent to streaming, monitoring with EPICS, real time processing with CPU and GPU using EPICS «FANOUT PVs».
- Preprocessing algorithm can be dynamically selected using PVs.



Basic FPSC user interface using EDM (EPICS) application

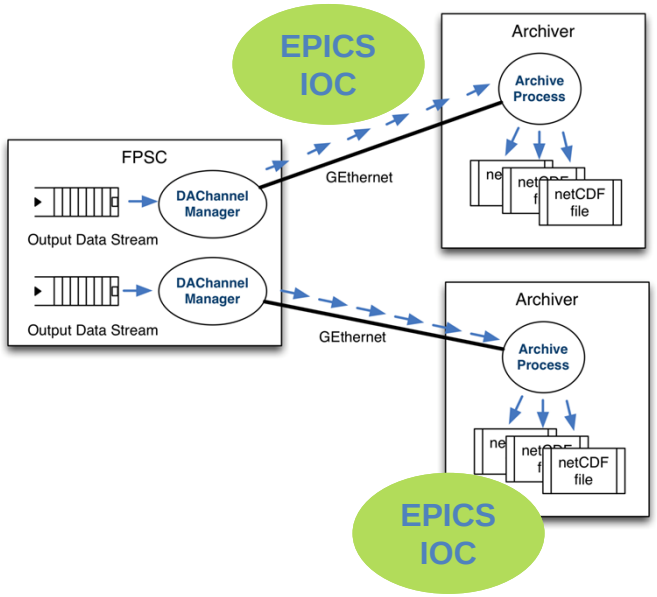






- Manual start/stop of FPSC
- Basic control of PVs during the pulse.
- Implementation of IocLog client in LabVIEW

Archiving System, Archiving Viewer and monitoring



- Data sources can be assigned to data archivers
- netCDF file is the fundamental storage unit
- A file per data source (signal) and pulse
- Two types of data are currently implemented: "d1wave" and "event".
- "Online" and "Offline" mode
- On remote via NFS (Network File System)
- Time slice positioning
- Self Description data visualization
- Flexible plotter (Zooms, Export options
- Completely based on EPICS channel access
  - Every archiver implements its own EPICS IOC
- System variables: CPU load, Memory Usage
- Archiving system performance
  - Receiving data rate per channel
  - Total received data rate
  - Storing data rate per channel
  - Total saved data rate

O8-7 Rodrigo Castro
NetCDF based data archiving system proposal for the ITER Fast Plant System Control prototype

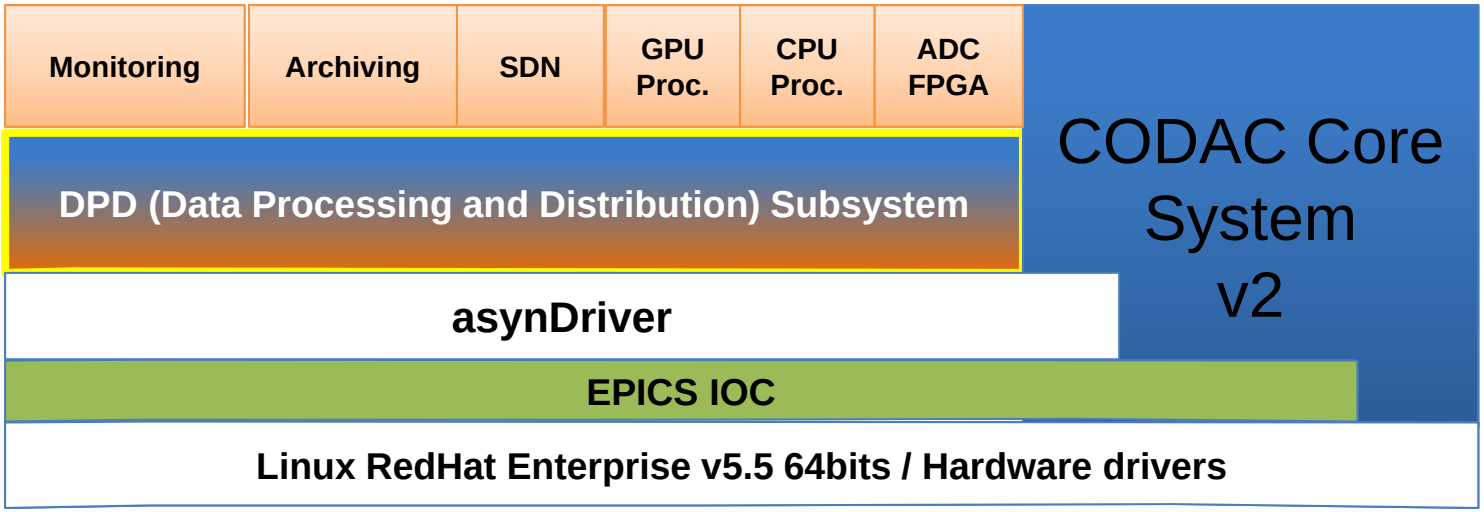
Results & Conclusions for Alpha version

- Implementation of a FPSC devoted to data acquisition following main ITER requirements:
  - Data acquisition using FPGA DAQ devices with IEEE1588 hardware time-stamping (PXI-6682).
    - Sampling rate limited by time-stamping method (≈100kS/s per channel).
  - System DAQ parameters controlled by EPICS PVs (changed dynamically during the PULSE)
  - Data movement and distribution controlled by EPICS PVs.
    - Data flow controlled by EPICS .
  - Streaming/archiving capabilities implemented with NETCDF.
  - Preprocessing algorithms controlled by EPICS PVs and executed in the local processor and/or the GPU.
  - Integration with EPICS CODAC CORE SYTEM V1.1.
- LabVIEW based tools (RT/FPGA) were a good choice for quick prototyping, modeling, and testing in a short period of time (3 months).

FPSC PXIe form factor: objectives for beta version

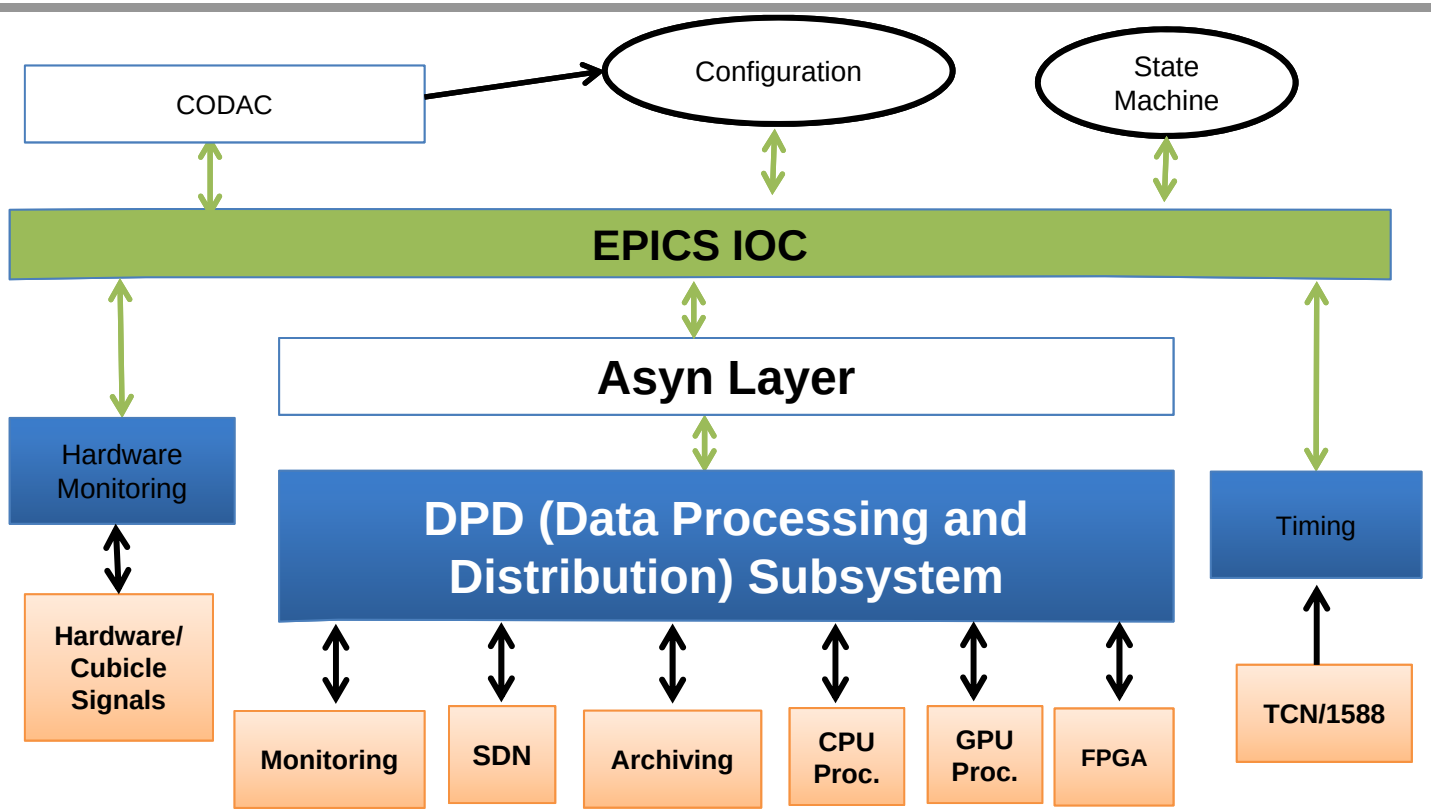
- Hardware:
  - Use of COTS hardware (PXI chassis with external controller, timing modules, DAQ using FlexRIO)
  - Integration of the GPU in the controller.
  - Integration in an ITER standard cubicle.
- Software:
  - Use of CODAC Core System V2 (RHEL 5.5 64 bits)
  - Use of EPICS base 3.14.12 and Asyndriver 4.16
  - FPSC software has to include different "functional elements" in its architecture: EPICS monitoring, data preprocessing, data acquisition with FPGA solutions (asyn NI-RIO driver), remote archiving, etc.
  - FPSC software implementation must be totally integrated with both EPICS and asynDriver technologies.
  - FPSC software has to move the data acquired (at high sampling rates) with good performance and reasonable resources (no more than 50% CPU load).
  - FPSC core software has to manage different data types
  - FPSC core software has to provide fault tolerant mechanisms for data movement.

Complete software architecture of the FPSC prototype: beta version



- Model based on:
  - Concurrent "functional elements" (threads).
  - Very efficient data block transmission among threads, avoiding not only locks but also memory allocations and interrupts.
  - Data transmission based on data blocks that encapsulate any data type.
  - Completely integrated with EPICS and asynDriver.
  - Data movement based on standard *"epicsRingBuffer"* elements (included in EPICS base distribution).

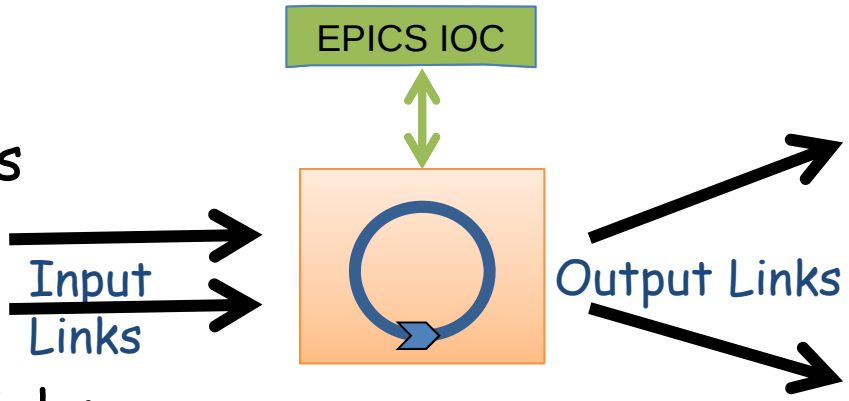
Data Processing and Distribution



- The core of FPSC software (the DPD) allows for:
  - Moving data with very good performance.
  - Integrating all the functional elements (EPICS monitoring, Data processing, Data Acquisition, Remote archiving, etc).
  - Having a code completely based on the standard asynDriver.
  - Full compatibility with any type of required data.

DPD features

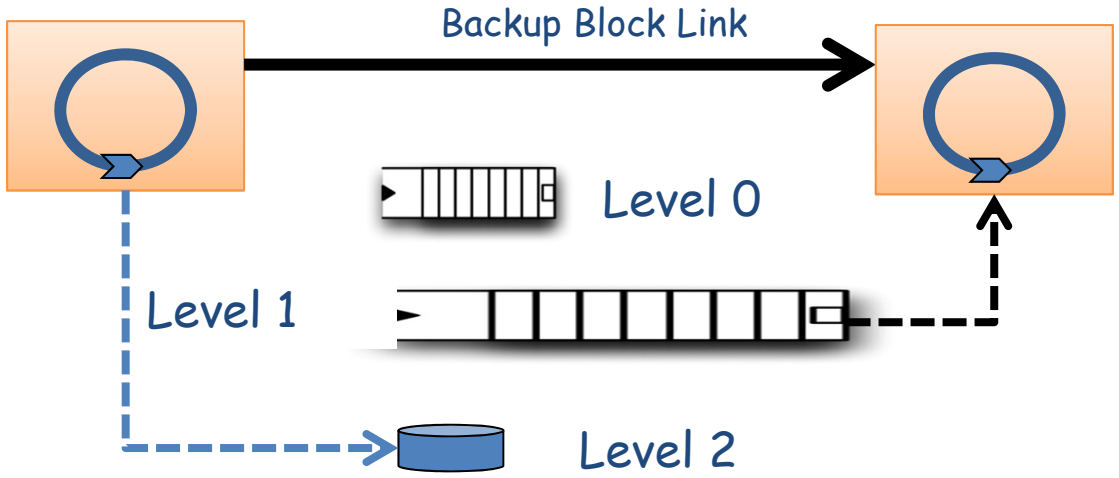
- DPD enables to configure both the different functional elements (FPGA acquisition, SDN, EPICS monitoring, data processing, data archiving) of the FPSC and the connections (links) between them.
- Functional elements allow:
  - reading data blocks from inputs
  - processing received data
  - generating new signals
  - routing data blocks to output links



- DPD enables the integration of new type of functional elements to extend the FPSC functionality. This implies the creation of the corresponding asynDrivers that can be carried out in a simple way.
- Enables a very easy integration of any existing asynDriver.

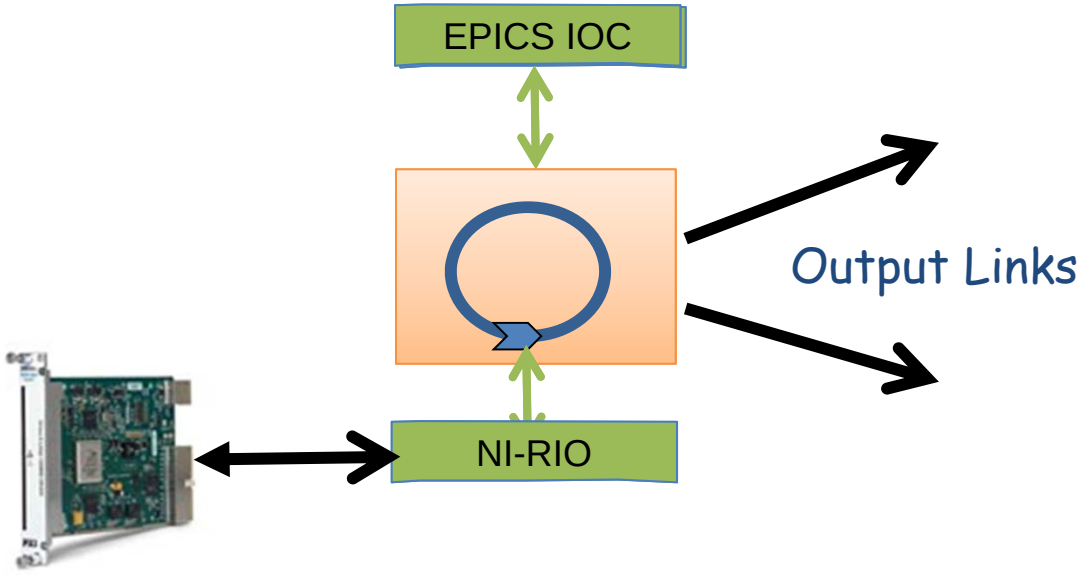
DPD features

- DPD enables to configure the data routing at configuration-time or even at run-time (to implement fault tolerant solutions).
- DPD provides a common set of EPICS PVs for the several functional elements and their respective links.
- DPD provides on-line measurements of both throughputs and buffer occupancy in the links.
- DPD implements an optional multi-level buffering (memory, disk) backup solution for any link of the system.



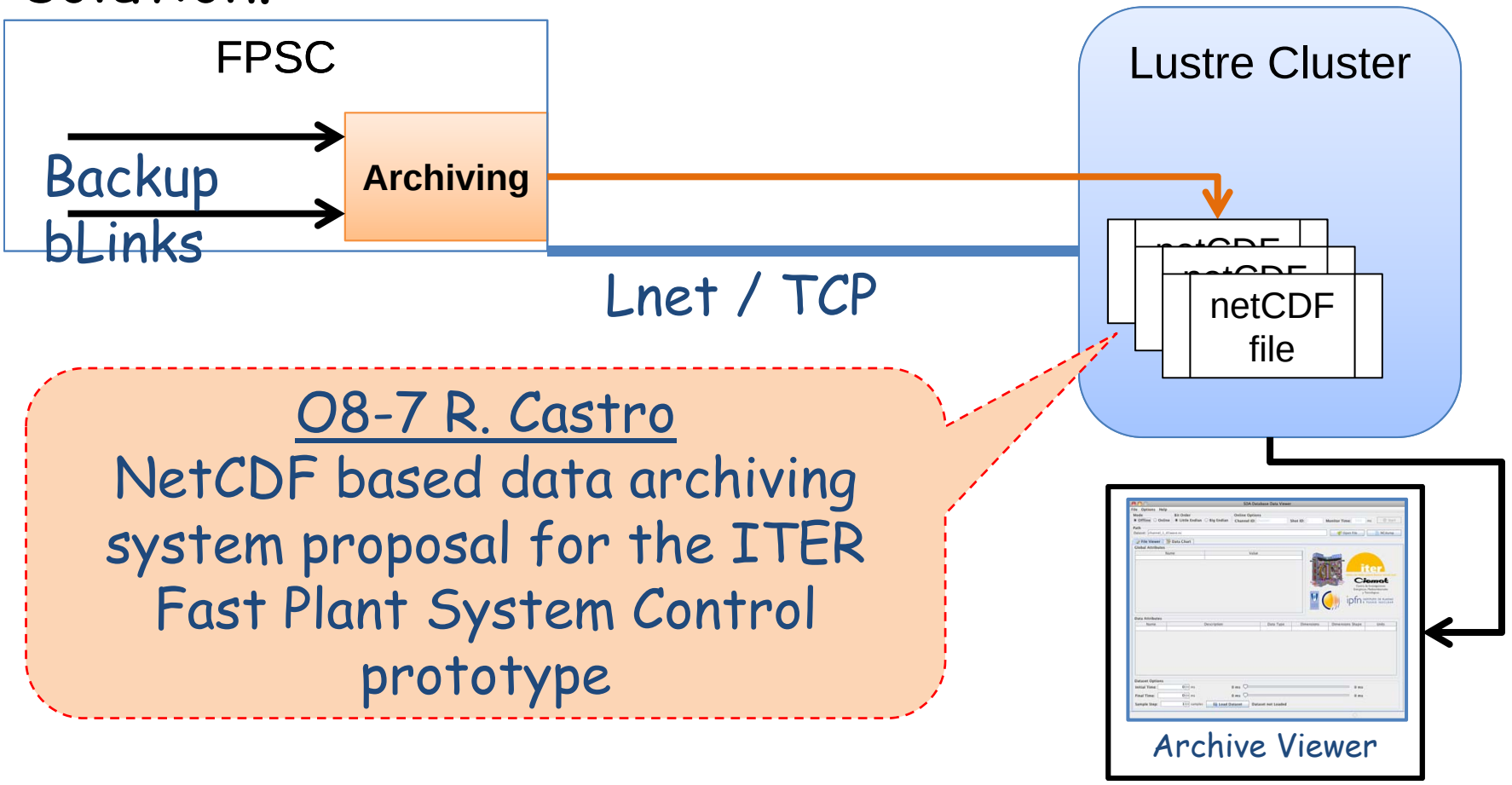
Example of functional block: Asyn FPGA NI-RIO

- Standard asyn driver for acquiring data using NI-RIO devices.
  - Int32, arrayfloat32 (DMA channels with waveforms)
- Implementation of a generic model to interface asyn driver with NI-RIO FPGA.
  - General guidelines for modeling the implementation of the RIO device code using LabVIEW/FPGA.
  - Standardization of information included in the DMA channels.



More functional blocks: Archiving Solution

- FPSC uses an asyn module connected to a Cluster solution.



Results: DPD data acquisition performance

Desktop PC

ITER Catalog

CPU
Intel I3 540
3.07GHz

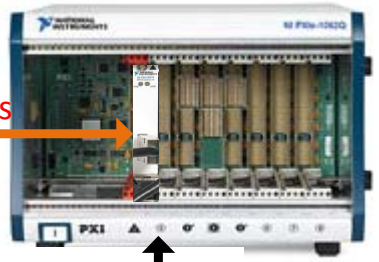
H55
Intel Chipset

PCIe x16

PCIe-8372

PCIe x4 566MB/s 528MB/s

PXIe Chassis



PCIe x4 1GB/s 528MB/s

NI5752

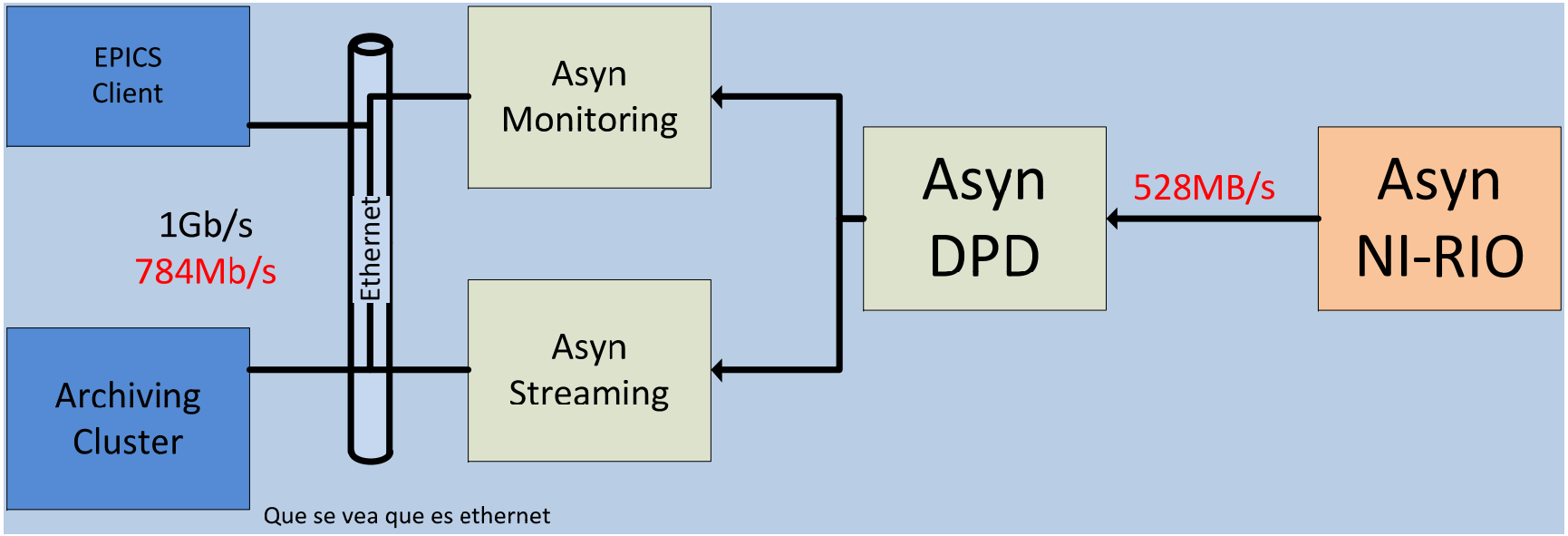
FlexRIO Device
PXIe-7965R

32 channels
Sampling rate:
6MS/s

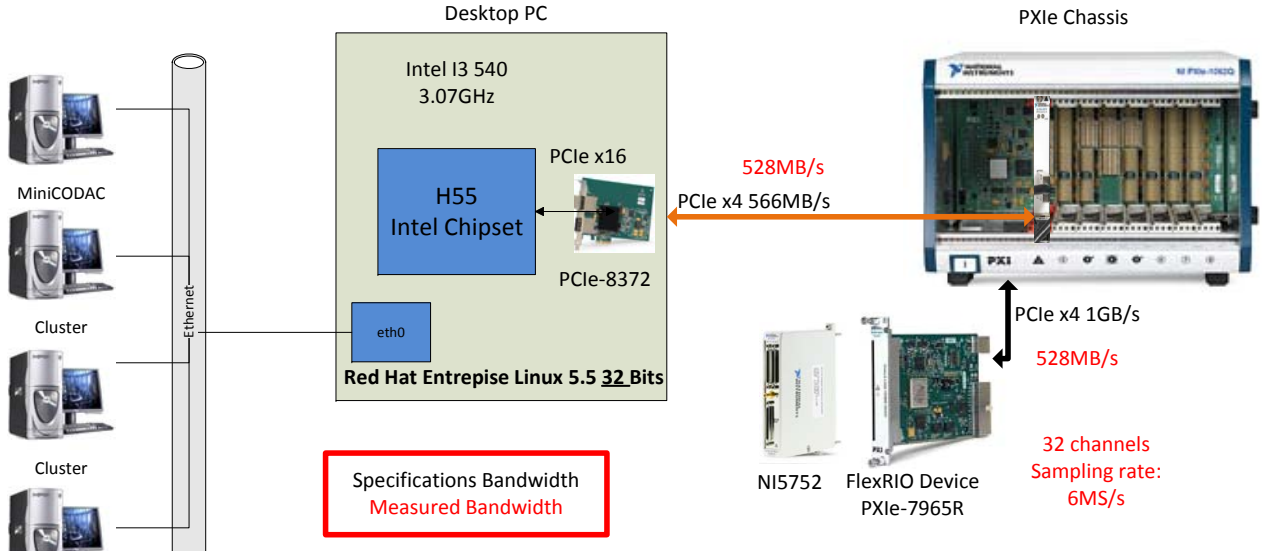
Red Hat Entrepise Linux 5.5 32 Bits

Specifications Bandwidth
Measured Bandwidth

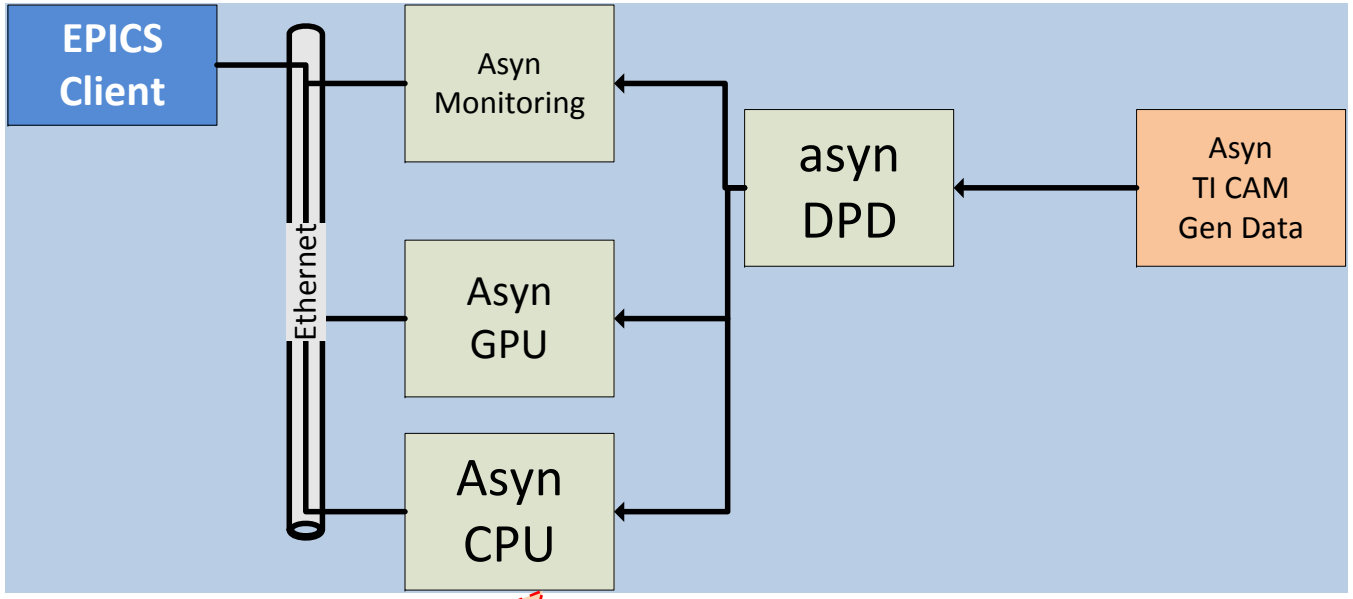
DPD with data acquisition, archiving channels and monitoring with EPICS



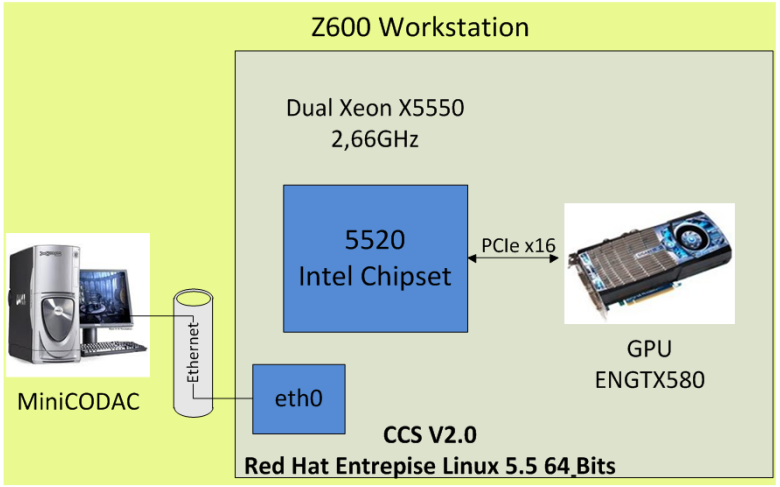
Cpu load =35%



DPD with CPU & GPU processing data and EPICS monitoring



Poster P2-11 J.Nieto
A GPU-based real time high performance computing service in a fast plant system controller prototype for ITER



Conclusions for the PXIe form factor beta version

- An ITER CODAC FPSC prototype has been implemented.
- FPSC software implementation is totally integrated with EPICS and asyndriver technology.
- We have obtained a prototype of DPD that is able to include different functional elements in its architecture: EPICS monitoring, data processing, Data Acquisition (asyn NI-RIO driver), GPU, Remote archiving, etc.
- The DPD is the FPSC core. It is capable of moving acquired data with good performance and reasonable resources (low CPU load).
- The FPSC core is able of managing different types of data
- Fault tolerant mechanisms are provided in the interconnection of the different functional elements.